JDCS
디지털콘텐츠학회논문지
Journal of Digital Contents Society
Vol. 23, No. 9, pp. 1813-1822, Sep. 2022

Check for updates

하드웨어 기반 암호화와 메모리 격리를 통한 경량 RISC-V 신뢰 실행 환경

박 우 정[1] · 김 해 용[1] · 지 장 현[1] · 문 재 근[1] · 김 호 원[2,3*]
[1]스마트엠투엠 기업부설연구소 연구원 [2]스마트엠투엠 기업부설연구소 연구소장 [3*]부산대학교 전기컴퓨터공학부 교수

Lightweight RISC-V Trusted Execution Environment with Hardware-based Encryption and Memory Isolation

Woojung Park[1] · Haeyoung Kim[1] · Janghyun Ji[1] · Jaegeun Moon[1] · Howon Kim[2,3*]

[1]Researcher, R&D Center, SmartM2M, Busan 48058, Korea
[2]Technical Director, R&D Center, SmartM2M, Busan 48058, Korea
[3*]Professor, Department of Computer Engineering, Busan National University, Busan 46241, Korea

[요 약]

신뢰 실행 환경(trusted execution environment)은 민감한 데이터와 코드를 비신뢰 환경으로부터 격리하여 외부 탬퍼링에 대한 저항성을 제공하고 안전한 실행을 보장하기 위한 기술이다. 하지만, 기존의 신뢰 실행 환경 기술은 경량 임베디드 환경의 외부 메모리에 대한 물리적 공격 완화, 실시간 작업을 안전하게 격리하기 위한 메모리 격리 규칙에 대한 보호, 다중 신뢰 영역과 같은 기능의 지원이 미비하다. 본 논문에서는 최근 많은 연구가 이루어지고 있는 저전력 RISC-V 기반 오픈 소스 프로세서 구현인 Ibex를 기반으로 다중 신뢰 영역을 안전하게 격리하여 동시에 실행할 수 있는 신뢰 실행 환경 구조를 제시한다. 본 구조는 온 칩(on-chip) 하드웨어 암호 가속기를 통해 데이터 및 코드의 기밀성과 무결성을 효과적으로 보장하며 고도화된 소프트웨어 및 물리적 공격에 대한 높은 저항성을 가진다.

[Abstract]

A trusted execution environment (TEE) is a technology that isolates sensitive data and code from untrusted environments to provide resistance to external tampering and ensure safe execution. However, the existing trusted execution environment technologies lack support for functions such as mitigation of physical attacks on external memory in a lightweight embedded environment, protection of memory isolation rules for safely isolating real-time tasks, and multiple trusted domains. In this paper, based on Ibex, a low-power RISC-V-based open-source processor implementation that has been widely researched in recent years, we present a trusted execution environment structure that can safely isolate and execute multiple trusted domains at the same time. Our structure can ensure confidentiality and integrity of data and code efficiently through on-chip hardware cipher accelerator and has high resistance to advanced software and physical attacks.

색인어 : RISC-V, 신뢰 실행 환경, 메모리 암호화, 단일 칩 시스템, 물리적 메모리 보호

Keyword : RISC-V, Trusted Execution Environment, Memory Encryption, System on a Chip, Physical Memory Protection

http://dx.doi.org/10.9728/dcs.2022.23.9.1813

Received 28 July 2022; **Revised** 16 August 2022
Accepted 18 August 2022

***Corresponding Author; Howon Kim**
Tel: ※ 개인정보 표시제한
E-mail: howonkim@gmail.com

Ⅰ. 서 론

시스템 반도체 기술의 발전과 사물인터넷 기술의 확산으로 임베디드 시스템이 다양한 분야에서 활용되고 있다. 특히 스마트 공장과 스마트 시티와 같이 주요 인프라 제어 및 관리에 활용되는 IoT 디바이스가 증가함에 따라, 경량 임베디드 시스템에 대한 보안 위협과 파급력 또한 증가하고 있다. 이러한 시스템에서 소프트웨어 취약성을 이용하여 내부 민감 데이터에 접근하거나, 코드 위/변조를 통해 비정상적인 동작을 수행하게 하는 공격을 방지하기 위해 다양한 연구가 진행되고 있으며, 그 중 많은 분야에 적용되는 기술로 신뢰 실행 환경 기술(trusted execution environment)이 있다[1].

신뢰 실행 환경 기술은 신뢰된 영역의 데이터와 코드를 비신뢰 환경으로부터 격리하여, 잠재적으로 적대적인 환경에서도 민감 데이터와 코드를 보호하고 안전한 실행을 보장한다. 이는 크게 메모리에 대한 접근 제어를 이용한 격리 기법만을 활용하는 기술과, 하드웨어 기반 메모리 암호화를 통해 물리적인 격리 또한 제공하는 기술로 구분하여 논의할 수 있다.

ARM TrustZone은 모바일 환경에서 가장 널리 사용되는 신뢰 실행 환경 기술이다. 해당 기술은 시큐어(secure) 영역과 노멀(normal) 영역으로 구분된 두 개의 실행 환경을 제공하며, 노멀 영역에서 시큐어 영역으로 접근을 방지하여 메모리상의 주요 데이터를 보호한다[2]. 하지만, 이는 메모리에 대한 접근 제어를 이용한 격리 기법만을 활용하여 메모리에 대한 물리적인 공격 혹은 분석에 대한 보호 수준이 제한되어 있다. 또한, 실행 환경이 두 개로 제한되어 다중 신뢰 영역간의 격리가 어렵다.

한편, 임베디드 환경에서는 공격자의 물리적인 접근이 가능하여 외부 메모리에 대한 물리적인 보호가 필요하다. 이에 대한 예시로, 무기 체계를 작동시키는 프로그램은 무기가 공격자에게 탈취되었을 경우 프로그램의 작동 중에 수집, 처리하는 데이터뿐만 아니라 프로그램의 코드 자체도 노출 방지 및 보호가 적용되어야 한다. 이 때, 프로세서 칩 외부 메모리에 평문 데이터 또는 코드가 존재할 경우 기기에 물리적으로 접근이 가능한 공격자에게 기밀 내용이 노출될 수 있다. 따라서 외부 메모리에 존재하는 데이터와 코드를 암호화하여 보호할 필요가 있다.

인텔 SGX (Software Guard eXtension)는 데스크톱과 서버 환경에서 널리 사용되는 인텔 아키텍처의 확장으로, 메모리에 대한 물리적인 보호를 제공하는 신뢰 실행 환경 기술이다. 이는 프로세서의 메모리 암호화 엔진(memory encryption engine)을 통해 메모리 접근을 암호화한다[3]. 하지만 이는 기반이 되는 프로세서가 고성능 컴퓨터 시스템에서 작동하여 경량 임베디드 환경에서 활용하기가 어렵다.

이와 같이, 기존의 신뢰 실행 환경 기술은 연산 능력, 메모리, 에너지가 제약된 경량 임베디드 환경에서는 메모리 암호화를 지원하지 않는 경우가 많다. 이는 경량 환경에서 암호화를 지원하기 위한 비용이 크기 때문에 효율성의 문제가 발생하는 것과, 이로 인해 새로운 아키텍처에 대한 수요가 한정적인 것이 주요 원인으로 볼 수 있다. 이를 해결하기 위해서는 효과적인 성능을 지니고, 기존 프로세서를 활용하여 최소한의 아키텍처 변경을 통해 적용이 가능한 메모리 암호화 기술이 요구된다.

또한, 임베디드 환경에서는 다양한 센서 데이터 처리와 연산에 대한 작업(task)이 동시에 이루어지는 경우가 많다. 이러한 환경에서는 작업이 끝까지 실행되지 않은 경우에도 문맥 전환(context switching)을 통해 다른 작업을 실행할 수 있어야 하고, 특정 작업이 소프트웨어 취약성을 공격받았을 경우 침해 범위가 제한될 수 있도록 개별 작업을 격리하여 실행할 필요가 있다. 이를 위해서는 동시성과 다중 실행 환경이 지원되는 메모리 격리 기술이 필요하다.

다중 실행 환경이 지원되는 메모리 격리 기술을 구현하기 위해서는 실행 환경에 대한 전환이 발생할 때 기반 하드웨어에서 지원하는 메모리 격리 규칙을 변경하는 방법이 있다. 다중 실행 환경에 이와 같은 방법을 적용하면 신뢰 실행 환경의 개수에 대한 제약이 없고 유연한 격리 설정이 가능한 것을 장점으로 지니지만, 공격자가 소프트웨어 취약점을 통해 메모리 격리 규칙의 변경을 위한 권한을 획득하여 규칙을 변조하는 것을 막기 어려운 단점이 있다. 이러한 공격을 방지하기 위해서는 메모리 격리 규칙에 대한 별도의 무결성 검증을 수행할 필요가 있다.

본 논문에서는 최근 많은 연구가 이루어지고 있는 저전력 RISC-V 기반 오픈 소스 프로세서 구현인 Ibex를 기반으로 하여, 경량 임베디드 환경에서 외부 메모리에 대한 보호, 동시성 및 다중 실행 환경, 메모리 격리 규칙에 대한 보호를 지원하는 신뢰 실행 환경 구조를 제시한다. 이를 위해 하드웨어 암호 가속기를 활용해 효과적으로 메모리를 암호화할 수 있는 SoC를 구성한다. 그 과정에서, 머클 트리(merkle tree)를 활용하여 메모리 영역에 대한 위/변조와 재전송 공격(replay attack)을 효율적으로 검증한다. 또한, RISC-V의 PMP (Physical Memory Protection) 기능과 확장 PMP를 활용하여 작업 간의 문맥 전환이 발생할 때 접근 가능한 메모리 영역을 제어하고, M-모드(machine mode)에서 동작 중인 커널로부터 U-모드(user mode)에 대한 메모리 격리를 적용한다. 이에 더하여, 하드웨어 암호 가속기의 AEAD (Authenticated Encryption with Associated Data) GCM (Galois/Counter Mode) 운영 모드와 캐시에 대한 접근 제어를 활용하여, 활성 작업 영역에 대한 PMP 규칙의 무결성을 검증한다. 이를 통해 다중 작업을 안전하게 격리하여 동시에 실행할 수 있고, 데이터 및 코드의 기밀성과 무결성을 보장할 수 있는 시스템을 구성한다.

본 연구의 구성은 다음과 같다. 2장에서는 본 연구와 관련된 배경지식을 설명한다. 3장에서는 신뢰 실행 환경을 위한 설계의 주안점, 설계 내용, 취약점에 대한 대응 내용을 설명한다. 4장에서는 본 연구와 기존 연구와의 차별점을 설명하며 5장에서는 결론과 향후 연구를 제시한다.

II. 배경지식

2-1 RISC-V

RISC-V는 UC 버클리에서 개발되었으며 사용이 자유롭고 공개된 ISA (Instruction Set Architecture)이다. 기존의 ISA는 대부분 증분형 (incremental) ISA로 상업성을 위해 하위 호환성을 유지하여 왔기 때문에, 자주 사용되지 않는 명령어를 포함시키기 위해 성능을 희생해야 하는 문제가 존재하였다[4]. 이에 비해, RISC-V는 모듈형(modular) ISA로, 기본 ISA에 해당하는 RV32I를 기반으로 하여 필요에 따라 다양한 표준 확장을 구현한다[5]. 이는 프로세서의 설계 목적에 적합한 명령어 집합 모듈을 선택적으로 적용하여 자원을 효율화하는 것이 가능한 장점이 있다.

RISC-V는 개방적이고 효율적인 구조로 인해 많은 연구와 프로세서 개발이 이루어지고 있다. 상용 프로세서인 SiFive 사의 코어 IP와 오픈 소스인 Ibex 등이 대표적이다. 특히 Ibex는 오픈 소스 RISC-V 코어 중 최초 Zero-riscy라는 명칭으로 오픈 소스 RISC-V 아키텍처인 PULP 플랫폼에 포함되어 개발된 프로세서이다. 초저전력 프로세서 코어로서 우수한 전력 대 성능비를 가지며 생산에 사용이 가능한 등급(production-quality)임을 강점으로 내세우고 있다[6], [7], [8]. 또한, 부채널 공격에 대한 대응, M-모드에 대한 추가적인 접근 제어를 위한 확장 PMP와 같이 보안 기능들이 다수 적용되어 있다.

2-2 PMP (Physical Memory Protection)

PMP는 RISC-V 표준인 특권 사양(privileged spec)에 포함된 기능으로 지정된 각 영역에 대해 메모리 접근 권한을 설정하는 기능이다. RISC-V의 가장 높은 특권 수준(privilege level)인 M-모드에서 CSR (Control and Status Register)을 변경하여 설정이 가능하다. 최대 16개의 메모리 영역에 대한 읽기, 쓰기, 실행 권한에 따라 낮은 특권 수준의 접근을 허용하거나 거부할 수 있다. 이는 소프트웨어를 통해 정의가 가능하여 보호 도메인의 변경 및 관리가 유연한 장점이 있다.

한편, RISC-V 특권 사양에 따르면 M-모드는 무제한적인 접근 권한을 가진다. 따라서, 중간 수준인 S-모드가 존재하지 않고 M-모드와 U-모드만이 존재하는 저면적 RISC-V 구조에서 낮은 특권 수준을 높은 특권 수준으로부터 격리하기 위해서는 기능 확장이 필요하다. 이는 별도의 Smepmp 표준으로 정의되어 있다[9]. 또한, PMP는 RISC-V에서 신뢰 실행 환경을 구성할 때 메모리 격리를 위한 기술로 사용될 수 있다[10], [11], [12].

2-3 신뢰 실행 환경(TEE)

신뢰 실행 환경을 구성하는 방법은 다양하지만, 공통적인 관점으로부터 볼 때 이는 민감한 데이터와 코드를 비신뢰 환경으로부터 격리하여 외부 탬퍼링에 대한 저항성을 제공하고 안전한 실행을 보장하기 위한 기술로 볼 수 있다. 이는 암호화, 가상화, 하드웨어 기반 격리 등 다양한 방법으로 구현된다[10], [11], [13].

신뢰 실행 환경의 대표적인 구현에는 ARM TrustZone이 있다. 이는 2004년에 ARM 프로세서에 도입된 하드웨어 보안 확장이다. ARM TrustZone은 전체 영역을 시큐어 월드(secure world)와 노멀 월드(normal world)로 구분하며, 각 영역에 대한 하드웨어 기반 격리를 수행한다[2].

신뢰 실행 환경에서 의미하는 비신뢰 요소에는 운영 체제(operating system) 자체가 포함된다. 소프트웨어를 이용할 경우 이러한 요소로부터 신뢰 요소를 격리하기가 어렵다[1]. 따라서, 신뢰 실행 환경에서 높은 보호 수준을 달성하기 위해서는 하드웨어를 이용한 격리 기술이 필수적으로 요구된다[1], [2], [10], [11], [12].

2-4 AEAD (Authenticated Encryption with Associated Data)

암호 알고리즘은 기본적으로 비밀 정보인 키를 통해 평문을 암호화하여 중요 정보를 보호하거나, 암호문을 복호화하여 원본 정보를 얻을 수 있는 방법을 제공한다[14]. 그 중에서도 AEAD는 메시지의 일부에 대해서는 기밀성이 보장되어야 하고 나머지 일부는 공개되어 있을 때 전체 메시지를 인증하기 위한 기술을 제공한다. 이는 예시로, 암호화 된 패킷이 전송될 때 암호문과 함께 패킷 헤더와 같은 추가적인 정보를 인증하기 위해 사용할 수 있다. 이는 암호문에 대한 인증만을 제공하는 AE (Authenticated Encryption)와는 차이가 있다[15].

블록 암호 알고리즘을 안전하게 이용하기 위해서는 사용 목적에 적합한 운영 모드(mode of operation)를 선택해야 한다. 최근에는 암호화와 함께 메시지 및 추가 데이터에 대한 인증이 가능한 AEAD 운영 모드들이 쓰이고 있으며, 특히 GCM 운영 모드는 NIST (National Institute of Standards and Technology)에 의해 표준화되어 많은 검증이 이루어진 운영 모드로 여러 메시지에 대한 병렬 연산이 가능하여 성능이 높은 장점을 가진다[16].

2-5 머클 트리(merkle tree)

머클 트리는 데이터의 검증을 위한 데이터 구조(data structure)로, 말단 노드가 대상이 되는 각 데이터 블록에 대한 해쉬 값으로 되어 있고, 각 노드에 대한 부모 노드가 자식 노드에 존재하는 해쉬 값들에 대한 해쉬로 구성된 트리이다. 이는 블록체인이나 스마트 그리드와 같은 다양한 환경에서 데이터 검증을 위한 목적으로 사용되고 있다[17], [18].

머클 트리는 각 말단 노드에 연관된 데이터가 변경되었을

경우, 변경된 말단 노드에서부터 루트 노드까지의 경로만을 재연산하여 전체 트리에 대한 재검증이 가능하다. 이를 통해 큰 데이터의 부분적인 변경에 대해 효과적으로 검증할 수 있다.

2-6 ECC (Elliptic-Curve Cryptography)

공개키 암호 알고리즘은 암호화를 위한 키와 복호화를 위한 키가 다른 암호 알고리즘으로, 키 교환이나 전자서명과 같은 목적을 위해 사용되고 유연성이 높은 것을 특징으로 한다. [19] 키 교환은 안전하지 않은 네트워크에서 두 명의 당사자가 공통의 비밀 키를 공유하기 위해 사용될 수 있고, 전자서명은 디지털 메시지의 무결성을 검증할 수 있다. ECC는 공개키 암호 알고리즘의 하나로, ECDH (Elliptic-Curve Diffie-Hellman)를 이용한 키 교환과 ECDSA (Elliptic-Curve Digital Signature Algorithm)를 이용한 전자서명이 가능하다.

Ⅲ. 본 론

3-1 설계 시 주안점

본 논문은 소프트웨어 기반 코드 및 데이터 암호화 실행 기술의 성능 문제를 해결함과 동시에, 배치 비용과 개발 비용을 최소화하기 위하여 메모리 격리 계층(memory isolation layer)과 버스 인터페이스 사이에서 동작하는 메모리 보안 모듈(memory security module)을 제시한다. 메모리 보안 모듈은 기존 메모리 공간을 암호화 실행 영역과 비암호화 실행 영역으로 구분하며, 프로세서의 메모리 접근 요청에 따라 암호화된 메모리의 데이터를 복호화하여 프로세서에 제공하거나, 암호화되지 않은 프로세서의 데이터를 암호화하여 메모리에 저장하는 역할을 수행한다. 경량 임베디드 환경에서 암호화 실행 영역 제공을 제공하기 위해 제시된 메모리 보안 모듈은 다음과 같은 요구사항을 고려하였다.

1) Enhanced Security

메모리 격리 기술은 소프트웨어 취약점을 이용해 비인가된 메모리 영역에 접근하는 공격을 방지하기 위한 기술이다. 하지만, 이는 디버그 인터페이스 혹은 외부 메모리에 가해지는 물리적 공격에 대한 보안을 제공하지 않으며, 공격자가 메모리 격리 규칙의 변경을 위한 권한 획득 시 규칙 변조를 통해 보안 기능을 무력화하는 것을 단독으로 방지하기 어렵다.

본 논문의 메모리 보안 모듈은 메모리 상의 암호화 실행 영역에 대한 기밀성과 무결성 제공을 통해 물리적 공격을 방어하며, 해당 영역의 메모리 격리 규칙에 대한 무결성을 검증하여 이에 대한 변조를 방지한다. 이를 위해 메모리 보안 모듈은 머클 트리와 결합된 GCM 운영 모드를 이용하고, 추가 인증 데이터와 L2 캐시에 대한 접근 제어를 통해 메모리 격리 규칙을 검증하여 암호화 실행 영역 메모리를 보호한다. 머클 트리와 결합된 GCM 운영 모드는 디버그 인터페이스 혹은 외부 메모리에 대한 물리적 공격을 방지할 수 있을 뿐 아니라, 높은 권한을 획득한 공격자가 메모리 격리 규칙을 변조할 경우에도 공격을 감지 및 차단할 수 있도록 설계되었다.

2) Processor-independent Memory Isolation

최소한의 프로세서 내부 구조 변경으로 암호화 실행 영역과 메모리 격리 규칙에 대한 검증 기능을 제공하기 위해, 메모리 보안 모듈은 프로세서의 메모리 격리 계층과 버스 인터페이스 사이에서 동작한다. 이는 기존 SoC (System on a Chip)의 버스 인터페이스 상에서 간단한 모듈을 추가하고, 프로세서 내부의 특권 수준과 메모리 격리 규칙에 대한 외부 출력 인터페이스만을 메모리 보안 모듈에 연결하면 암호화 실행 영역 및 메모리 격리 규칙 검증을 제공할 수 있는 확장성을 제공한다. 본 기술은 Intel SGX와 같이 별도의 명령어가 필요하거나 격리 대상의 소프트웨어 변경을 요구하지 않기 때문에 배치 비용과 개발 비용을 최소화할 수 있다.

3) On-the-fly Secure Region Access

암호화 실행 영역의 암복호화 동작 수행은 프로세서의 직접적인 개입을 배제한 on-the-fly 방식으로 수행된다. 이를 위해 메모리 보안 모듈은 L2 캐시와 결합되어 제공된다. 암호화 실행 영역으로 설정된 메모리 영역은 L2 캐시의 관리 단위로 암호화 되어 저장되며, 메모리 오프셋과 프로세서 내부 상태에 따라 각 암호화 매개변수 설정과 머클 루트 검증을 수행한다. L2 캐시와 결합된 암호화 실행 영역 관리 구조는 기존 HSM (Hardware Security Module) 또는 소프트웨어를 이용한 코드 암복호화 기법 적용 시 발생하는 특권 계층 (privileged layer)의 개입을 제거하는 동시에, 암복호화 연산에 따른 병목 현상을 감소시킨다.

4) Hardware-based Crypto Engine

모든 암복호화 연산은 메모리 보안 모듈 내부의 하드웨어 암호 모듈을 통해 수행된다. 하드웨어 암호 모듈은 암호화 동작 수행 과정에서 발생하는 성능 문제를 해결하는 동시에, 주요 보안매개변수의 저장 위치를 내부에서 관리할 수 있도록 설계되었다.

메모리 보안 모듈은 메모리의 암복호화를 위한 ARIA 블록 암호 키와 ARIA 키의 안전한 주입을 위한 ECDH, KBKDF 모듈을 제공하며, 외부 L2 캐시 동작과 연동하여 암복호화 및 검증 작업을 수행한다. 이에 대한 초기화 및 키 설정을 위한 접근은 시큐어 부트 과정에서만 허용되며, 이를 위해 내부 유한 상태 머신과 관리 레지스터를 제공하여 인가되지 않은 접근을 방지할 수 있다.

3-2 제안 설계 내용

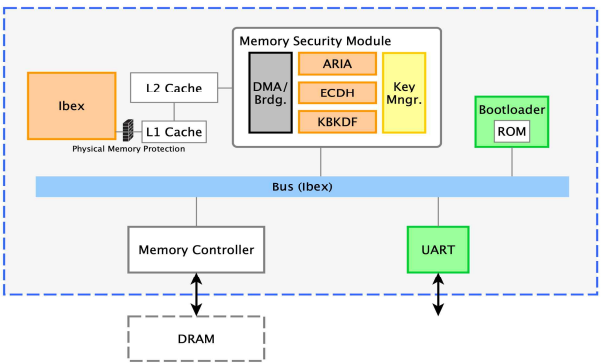

그림 1. 최상위 수준 SoC 구조
Fig. 1. Top level SoC structure

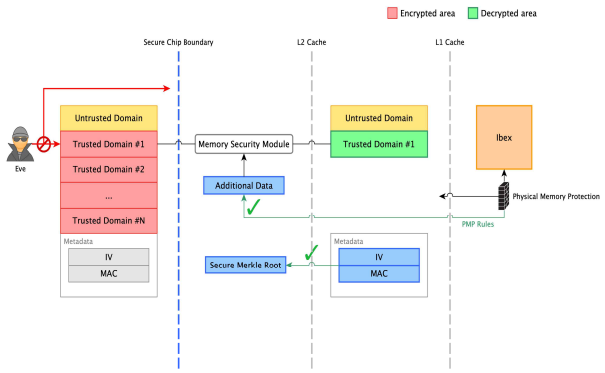

그림 2. 다중 신뢰 영역의 암호화 및 검증 구조
Fig. 2. Encryption and verification structure of multiple trusted domains

본 논문이 제시한 메모리 보안 모듈은 SoC 상의 RISC-V Ibex 프로세서와 버스 사이에서 암호화 인터페이스를 제공한다. 그림 1은 이를 위한 최상위 수준(top level) SoC 구조를 간략히 표현한다.

그림 1에 따르면, 암호화 인터페이스를 제공하기 위한 메모리 보안 모듈은 메모리 및 L2 캐시 영역에 대한 읽기와 쓰기를 수행하는 DMA(Direct Memory Access)/브릿지 모듈, 복호화를 수행하는 ARIA 암호 가속기 모듈, 키 교환 및 유도를 수행하는 ECDH 및 KBKDF 암호 가속기 모듈, 키 관리자 모듈로 세분된다. 메모리 보안 모듈은 키를 안전하게 주입하고 관리하며 메모리를 암복호화하기 위한 시스템의 생명 주기를 제공하며, 이를 통해 구성된 신뢰 실행 환경에서 각 세부 모듈은 각자 고유한 기능을 수행한다. 이 때, 효율적인 암복호화를 위해 메모리 보안 모듈이 복호화한 데이터는 L2 캐시에 기록된다. 또한, Ibex 프로세서의 모든 메모리 접근 시도는 RISC-V의 메모리 격리 기법인 PMP를 통해 적용된 규칙에 따라 보호 및 격리된다.

이외에도, SoC 상에는 부트로더, 통신을 위한 UART (Universal Asynchronous Receiver-Transmitter) 주변기기와 칩 외부에 존재하는 DRAM이 별도로 존재한다. 이는 칩 내부에 수용 가능한 데이터가 제한된 SoC 구조에서 칩의 부팅 과정에 필요한 최소한도의 인터페이스로, 물리적 공격이 가능한 칩 외부와 공격에 대한 완화가 적용된 칩 내부를 분리하여 시스템에 대한 공격 표면을 명확히 정의하고 최소화한다.

1) 메모리 암호화 및 보안 검증

공격자는 오프 칩 메모리에 대한 물리적인 공격을 통해 메모리에 대한 인가되지 않은 접근 권한을 얻을 수 있다. 이는 콜드 부트 공격(cold boot attack)을 통해 메모리의 일부 혹은 전체 값을 덤프하거나 DMA 공격을 통해 물리 메모리에 대한 완전한 읽기/쓰기 권한을 얻는 경우 등이 있다. 이러한 공격을 수행하면 공격자는 메모리에 존재하는 비밀 정보를 획득하거나, 시스템의 보안 정책을 우회하기 위해 설정을 변경할 수 있다.

본 논문이 제시하는 메모리 보안 모듈은 격리된 다중 신뢰 영역에 대하여 메모리를 암호화하고, 암호화 된 데이터와 메모리 격리 규칙에 대한 무결성을 보장하여 이에 대한 방어를 제공한다.

다중 신뢰 영역은 동시성의 단위인 작업을 기준으로 분할되어 암복호화되며, 각각의 작업은 독립적인 신뢰 영역이 된다. 활성 작업의 암복호화를 위한 주소 정보는 각 작업 간의 문맥 전환이 발생하는 시점에 변경된다. 이는 메모리 보안 모듈의 문맥 주소 레지스터(context address registers)에 기록되며 신뢰 영역의 코드 및 데이터에 대한 시작 주소와 끝 주소를 값으로 가진다. 코드를 실행하는 과정에서 메모리 접근이 발생하면 ARIA 암호 가속기를 통한 암복호화 및 검증이 수행되며 복호화 된 값은 L2 캐시에 기록된다.

신뢰 영역의 기밀성과 무결성을 보장하기 위해서는 각 영역의 현재 실행 중인 부분에 해당하는 코드 및 데이터가 AEAD 운영 모드를 통해 암호화되어야 한다. 이를 위해 ARIA 블록 암호의 GCM 운영 모드가 사용되며, 암호화된 이후에는 각 코드 및 데이터가 암호문과 검증을 위한 해시 값(인증 태그 값)으로 나누어진다.

운영 모드의 추가 인증 데이터는 위변조를 방지하기 위한 매개변수로, 활성 작업 영역의 PMP 규칙과 해당하는 암복호화 단위의 메모리 주소를 입력으로 가진다. PMP 규칙 인증은 신뢰 영역에 접근하기 위한 모든 메모리 연산에 사전에 정의된 고정 규칙이 반드시 적용되도록 하여, 공격자가 M-모드에 대한 무제한적인 코드 실행 권한을 획득하고 PMP 규칙의 변경이 가능한 경우에도 메모리 접근 공격을 방지한다. 이를 위해 메모리 보안 모듈은 프로세서 내부의 PMP 규칙에 대한 외부 출력 인터페이스를 입력으로 받는다. 또한, 메모리 주소 인증은 암호문이 존재하는 주소가 복호화 횟수에 관계없이 동일한 값으로 유지되어야 하는 특성을 활용하여, 각 암복호화 단위에 대해 메모리 주소 값의 변조를 통한 공격의 가능성을 사전에 차단한다.

활성 작업의 코드와 데이터는 L2 캐시에 복호화된 형태로 저장된다. 이는 프로세서에서 접근이 가능한 영역이므로, 커

널 및 새로운 활성 작업의 해당 영역에 대한 물리적인 접근 차단을 위해 신뢰 영역에 대한 M-모드의 L2 캐시 접근을 방지하고, 문맥이 전환되는 시점에 기존 L2 캐시에 대한 제거(eviction)를 수행한다. M-모드에서 신뢰 영역의 L2 캐시에 접근할 수 없도록 하기 위해, L2 캐시는 복호화 된 데이터를 구분하고 M-모드에서 이에 대해 접근할 수 없도록 하기 위한 간단한 접근 제어를 지원한다. L2 캐시에 대한 제거가 수행되면, 그 내부에 존재하는 데이터를 암호화하여 메모리에 기록하고 기존 값은 삭제한다.

한편, 암호문의 해시 값이 저장되는 DRAM은 오프 칩(off-chip)에 존재하기 때문에 공격자가 값을 변조할 수 있다. 이를 통한 재전송 공격을 방지하기 위해서는 검증을 위한 값을 별도로 저장해야 한다. 하지만, 코드와 데이터 전체에 대한 검증 값은 온 칩 하드웨어 내부에서 모두 관리하기 어렵다. 따라서, 전체 DRAM 영역은 코드 및 데이터 영역과는 별도로 검증을 위한 메타데이터 영역을 가지고, 핵심이 되는 값을 하드웨어 내에서 검증하여 전체에 대한 무결성을 보장한다.

메타데이터 영역에는 각 코드 및 데이터 암복호화 단위에 대한 해시 값으로 이루어진 머클 트리(merkle tree)가 저장되고, 하드웨어 내부에는 전체 트리의 검증을 위해 안전한 머클 루트(secure merkle root) 값이 저장된다. 또한, 암호화를 위한 넌스(nonce)가 별도로 메타데이터 영역에서 관리된다. 넌스 값은 GCM 운영 모드에서 매 암호화 시에 암호화를 위한 단위 영역 전체가 새로운 값으로 변경되어, 국소적인 데이터 변경에 대해 변경 위치를 추측하기 어렵도록 한다. 넌스 값은 고정(fixed) 부분과 변동(variable) 부분으로 구분되며, 변동 부분에는 카운터 값이 저장되어 매 암호화 시에 1을 더한 값으로 수정된다. 카운터 값의 중복을 방지하기 위해서는 이전에 사용된 값이 반복되지 않도록 충분한 크기의 비트 너비가 할당된다[20].

2) RISC-V PMP 규칙에 따른 메모리 격리

다중 작업이 동시에 존재하는 신뢰 실행 환경에서는 개별 작업, 혹은 시스템 콜을 제공하는 커널 영역이 공격받아 코드 또는 데이터에 대한 위변조가 발생할 경우에도 다른 작업 영역에 대한 기밀성과 무결성이 보장되어야 한다. 이러한 공격에 대한 완화를 위해서는 각 신뢰 영역에 대해 메모리 격리 규칙을 설정하여, 개별 작업이 잠재적으로 적대적인 외부 환경으로부터의 영향을 최소화하여 독립적으로 실행될 수 있도록 하여야 한다.

RISC-V 환경에서 이러한 신뢰 영역의 메모리 격리를 위한 방법으로, 보안 기능인 PMP를 사용할 수 있다. 이는 작업을 다른 신뢰 영역인 비활성 작업으로부터 격리하기 위한 PMP 규칙과 비신뢰 영역인 커널 영역으로부터 격리하기 위한 확장 PMP 규칙을 통해 각 신뢰 영역에 대한 메모리 격리를 적용한다. PMP 규칙은 문맥 전환이 발생하는 시점에 변경되는 규칙이며, 이에 대한 신뢰 영역은 동시성의 단위인 작업과 동일하게 정해진다.

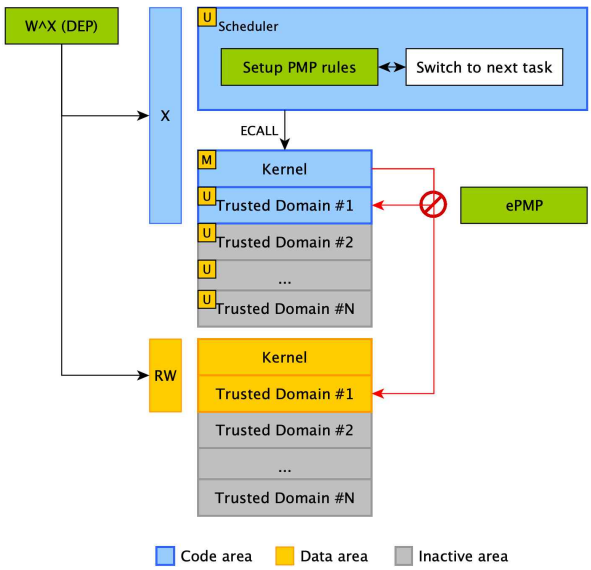

그림 3. 동시 다중 신뢰 영역에 대한 PMP 기반 메모리 격리 구조
Fig. 3. PMP-based memory isolation structure for concurrent multiple trusted domains

이에 비해, 확장 PMP 규칙은 부팅 이후 1회 설정되며 PMP 규칙에 대한 해석 방법을 변화시키는 규칙이다.

PMP 규칙은 활성 작업을 비활성 작업으로부터 격리하기 위해 현재 활성된 작업과 스케줄러 영역의 코드를 제외한 모든 코드가 U-모드에서 실행되는 것을 금지하도록 설정된다. 각 영역의 데이터는 활성 작업을 제외한 영역에 대한 읽기와 쓰기가 금지되도록 설정된다. 이는 RISC-V의 CSR인 pmpaddrX와 pmpcfgX를 변경하여 설정할 수 있다(X는 해당하는 레지스터의 번호). 동시성 보장을 위한 문맥 전환이 발생하면, 효율적인 문맥 전환을 위해 PMP 규칙 가운데 이전 활성 작업에 대한 규칙만이 현재 활성 작업에 대한 규칙으로 대치되며, 이는 시스템 콜을 통해 이루어진다.

한편, 비신뢰 영역인 M-모드의 커널로부터 U-모드인 신뢰 영역을 보호하기 위해서는 확장 PMP 규칙이 설정된다. 이는 RISC-V의 CSR 중 하나인 mseccfg의 MML (Machine Mode Lock) 및 MMWP (Machine Mode Whitelist Policy) 비트 설정을 통해 이루어진다. 이들 비트는 설정 시 리셋 이전까지 값이 유지된다. MML 비트 설정을 통해 U-모드에서만 접근 가능한 메모리 영역을 지정하는 것이 가능하며, MMWP 비트를 설정하여 지정되지 않은 전체 메모리 영역에 대한 M-모드 접근을 비허용할 수 있다. 각 비트 설정 시 기존의 PMP 규칙 해석 방식이 변화하기 때문에, 모든 PMP 규칙은 확장 PMP 규칙에 따라 작성되어야 한다.

PMP 규칙 설정 시, 각 코드 및 데이터 영역에 대해서는 데이터 실행 방지(data execution prevention)를 위한 W^X 규칙이 적용될 수 있다. W^X 적용 시 PMP 규칙에 대한 변경 없이는 공격자가 생성한 코드를 실행하는 것이 불가능하여 전체 소프트웨어가 코드 실행 공격에 대한 저항성을 가진다.

3) 암호 키 관리

시스템 상에서 관리가 필요한 키로는 키 교환을 위한 ECDH 공개키와 개인키, 고속 암복호화를 위한 ARIA 블록 암호 키가 존재한다. ECDH 공개키와 개인키는 부팅 과정에서 ARIA 블록 암호 키를 안전하게 주입하기 위한 키로, 유일한 키 쌍이 칩의 제조 과정에서 칩 내부에 각인된다. ARIA 블록 암호 키는 메모리에 대한 암복호화 수행에 필요한 키이며, ECDH 키 교환 및 KBKDF 키 유도 과정을 통해 매 부팅시마다 외부로부터 주입된다. 이 때, ECDH 키 교환의 결과는 타원곡선 위의 점이므로 균등 분포(uniform distribution)를 얻기 위해 KBKDF를 통한 키 유도를 사용한다[21]. 한편, 메모리에 대한 재전송 공격을 방지하기 위한 해시 값의 머클 루트 값 또한 키 관리 모듈이 별도로 관리한다.

시스템 외부에는 별도의 ECDH 공개키와 개인키가 존재한다. 이는 소프트웨어 프로그램 버전에 따라 변경되는 키로, 하드웨어 내의 메모리 암복호화를 위한 ARIA 블록 암호 키가 프로그램 버전에 따라 변경되어 소프트웨어의 변경 시에도 시스템을 안전하게 재사용할 수 있도록 한다.

메모리 보안 모듈 내에 존재하는 키 관리 모듈은 이러한 키에 대한 전반적인 주입, 저장, 출력 기능을 수행한다. 이 때, 해당 모듈은 칩 경계를 기준으로 할 때 칩 공개키에 대한 외부 출력만을 지원하며, 칩 개인키와 ARIA 블록 암호 키에 대한 외부 출력은 지원하지 않는다. 이는 칩 외부에서 코드 및 데이터와 같은 비밀 정보를 암호화하기 위해 사용된 키를 획득하기가 어렵도록 한다. 이를 통해 해당하는 정보에 대한 기밀성과 무결성을 보장한다.

3-3 취약점 대응

소프트웨어 취약점으로 인해 특정 신뢰 영역에 대한 침해가 발생한 경우, 공격자가 격리된 신뢰 영역을 탈출하기 위해서는 PMP 규칙을 변경해야 한다. 작업의 특권 수준은 U-모드이므로 이를 수행하기 위해서는 시스템 콜을 호출해야 하며, 반환 지점에 PMP의 코드 실행 권한이 표기되어 있어야 하므로, 특정 작업이 스스로의 PMP 규칙에 대한 변경을 시도하면 예외가 발생하여 침해 범위가 제한된다.

커널 영역에 대한 취약점으로 인해 침해가 발생하였을 경우 공격자는 신뢰 영역의 작업에 직접 접근할 수 없다. 접근을 위해서는 PMP 규칙의 변경 권한이 요구되지만, MML 비트 설정 시의 PMP 규칙은 M-모드에서 실행이 가능한 모든 코드 영역에 대한 잠금(lock)이 필요하도록 설계되어, M-모드에서 실행이 가능한 코드를 변조하여 PMP 규칙 변경을 수행하는 것은 불가능하다. 단, PMP를 설정하기 위해서는 특수한 CSR의 값을 변경해야 하고, 커널 영역은 비신뢰 영역으로 암호화되어 있지 않기 때문에 해당 코드에 대한 코드 재사용 공격(code reuse attack) 및 외부 메모리에 대한 물리적인 공격에는 대비할 필요가 있다. 이러한 공격이 발생할 경우에도, 본 시스템에서는 메모리 암호화 및 보안 검증 구조가 PMP 규칙에 대한 검증을 통해 신뢰 영역에 대한 격리를 유지하여 안전성을 보장하는 것이 가능하다.

Ⅳ. 기존 연구와의 비교

본 논문에서 제안하는 신뢰 실행 환경은 경량 임베디드 시스템에서 합리적인 비용으로 다중 신뢰 환경과 코드 및 데이터의 암호화, 메모리 격리 규칙에 대한 보호를 지원하여 물리적 공격 및 고도의 소프트웨어 공격에 대한 높은 보안성을 달성하는 것이 가능하게 하는데 그 요점이 있다. 기존의 신뢰 실행 환경에 대한 결과물이나 연구는 대부분 이를 부분적으로만 지원하고 있다. 본 장에서는 그러한 기존의 결과나 연구 중에서도 대표적인 ARM TrustZone, 인텔 SGX, KeyStone과 본 논문에서 제안한 구조를 비교하고자 한다.

ARM TrustZone은 모바일 환경에서 널리 사용되는 신뢰 실행 환경 기술로 시큐어 영역과 노멀 영역으로 구분된 두 개의 보호 도메인을 중심으로 기능한다[2]. 이러한 기반 아키텍처는 하드웨어 비용이 낮은 장점이 있지만, 다중 신뢰 영역간의 물리적인 격리 능력이 한정되어 있다. 또한, ARM TrustZone은 하드웨어에 대한 공격을 방지하기 위한 방법으로 DMA에 대한 접근 제어 등을 택하고 있다. 하지만, 이는 메모리에 대한 암호화를 적용하지는 않기 때문에 칩 외부 메모리에 대한 버스 탭 공격(bus tapping attack)과 같은 물리적 공격에는 취약하다[22]. 이에 비해 본 논문에서 제시한 아키텍처는 RISC-V의 표준 기능인 PMP 규칙에 대한 변경을 통해 다중 신뢰 환경을 지원하며 메모리 암호화를 통해 외부 메모리에 대한 물리적 공격으로부터 시스템을 보호한다.

인텔 SGX는 보안상 민감한 계산 작업에 대한 기밀성과 무결성을 제공하기 위한 인텔 아키텍처의 확장으로, 메모리 암호화 엔진을 통해 DRAM에 대한 보호 기능을 제공한다. 이는 다중 신뢰 환경을 지원하며 물리적 공격에 대한 보호를 제공한다[3]. 하지만 인텔 SGX는 데스크탑 혹은 서버와 같은 고성능 컴퓨터 시스템을 목표로 개발된 아키텍처로, 경량 임베디드 환경에서 활용하기가 어렵고, 또한 투기적 실행(speculative execution)으로 인한 공격 방법이 존재하는 등 복잡한 기반 아키텍처가 보안 공격의 대상이 되고 있다[23]. 이와 달리 본 연구에서 제시하고 있는 아키텍처는 경량 암호와 하드웨어 암호 가속기를 이용해 경량 임베디드 환경에서 사용하기가 적합하며, 최소한의 프로세서 변경을 통해 기능하여 하드웨어 개발 및 배치 비용을 최소화한다. 이러한 경량 임베디드 아키텍처는 단순한 내부 구조를 가지므로 비순차적 실행(out-of-order execution)이나 투기적 실행으로 인한 캐시 부채널 공격에서 비교적 자유롭다.

KeyStone은 RISC-V 환경에서 신뢰 실행 환경을 구축하기 위해 원하는 하위 기능 집합을 선택할 수 있도록 하는 오픈 소스 프레임워크이다. RISC-V 표준 기능을 사용해 구현

한 보안 모니터(security monitor)를 통해 메모리를 격리하며, 설정에 따라 하드웨어 기반 암호화를 적용하여 높은 성능으로 메모리를 암호화할 수 있다[11]. 하지만, KeyStone은 RISC-V의 S-모드(supervisor mode) 특권 수준을 통해 운영체제와 런타임 모듈이 동작할 것을 요구하여, Ibex와 같이 M-모드와 U-모드만이 존재하는 경량 RISC-V 시스템에서는 동작하지 않는다. 또한, 메모리 격리를 위해 최상위 특권 수준에서 동작하는 보안 모니터가 매번 설정하는 메모리 격리 규칙을 신뢰하기 때문에, 메모리 격리 규칙 자체에 대한 변조가 발생하는 경우가 별도로 보호되지 않는다. 이는 고도화된 소프트웨어 공격 및 물리적 공격에 대한 잠재적인 취약성을 내재한다. 반면에 본 논문의 시스템은 경량 RISC-V 시스템을 목표로 M-모드와 U-모드만을 이용하여 설계되어 베어메탈(baremetal) 시스템에서 효율적으로 동작하며, AEAD 암호를 활용하여 메모리 격리 규칙 자체에 대한 보호를 지원하여 메모리 격리 규칙 자체가 변조되었을 경우에도 공격을 탐지 및 방어하는 것이 가능하다.

V. 결론 및 향후 연구

본 논문에서는 저전력 RISC-V 프로세서인 Ibex 환경에서 하드웨어를 통한 메모리 암호화 및 PMP를 기반으로 한 메모리 격리를 이용하여 동시성을 지원하는 다중 신뢰 실행 환경을 구성하였다. ECDH 키 교환 및 KBKDF 키 유도를 활용하여 칩 외부로부터 안전하게 키를 주입하고 메모리를 암호화하였으며, 머클 트리를 이용하여 메모리에 대한 물리적인 재전송 공격을 방지하였다.또한, RISC-V의 보안 기능인 PMP와 확장 PMP를 통해 다중 신뢰 영역에서 동시에 실행 중인 작업에 대해 안전하게 메모리를 격리하였다. 그 과정에서 PMP 규칙에 대한 위변조가 발생할 경우 감지 및 차단할 수 있는 AEAD 암호화 구조와 캐시 접근 제어 구조를 설계하였다.

이러한 구조에서는 콜드 부트 공격과 DMA 공격과 같이 외부 메모리에 대한 물리적인 공격이 발생할 경우에도 기밀성과 무결성이 보장된다. 또한, 신뢰된 작업 영역 및 비신뢰 커널 영역에 대한 취약점으로 인해 침해가 발생할 경우에는 침해 범위가 제한되며, 시스템이 공격에 대한 저항성을 가짐을 보였다.

한편, 본 논문에서 제안한 구조에서는 부채널 공격에 대한 방어가 제공되지는 않는다. 본 구조는 매 부팅시 동일한 키와 타이밍을 통해 동작하여 부채널 공격에 대해 취약할 수 있다. 이를 해결하기 위해서는 부트 과정에서 전체 데이터와 코드를 매번 새로운 키를 통해 재암호화해야 하며, 각 하드웨어 요소가 부채널 공격에 대한 내성을 가져야 한다. 이에 대한 완화 방법은 향후 연구 과제로 남겨두었다. 또한, 본 논문의 설계는 원격 검증(remote attestation)과 같은 신뢰 실행 환경의 일부 기능을 제외하고, 최소한으로 정의된 하위 집합만을 포함한다. 이는 필요에 따라 추후 기능 확장을 통해 구현하는 것이 가능하며 이 또한 향후 연구를 위한 과제로 두었다.

감사의 글

본 연구는 국방과학연구소(계약번호: UD210023XD)의 지원에 의한 연구 결과임

참고문헌

[1] Jauernig, Patrick, Ahmad-Reza Sadeghi, and Emmanuel Stapf. "Trusted execution environments: properties, applications, and challenges." *IEEE Security & Privacy*, Vol. 18, No. 2, pp. 56-60, Mar 2020. https://doi.org/10.1109/MSEC.2019.2947124

[2] Pinto, Sandro, and Nuno Santos. "Demystifying arm trustzone: A comprehensive survey." *ACM computing surveys*, Vol. 51, No. 6, pp. 1-36, Nov 2019. https://doi.org/10.1145/3291047

[3] V. Costan and S. Devadas, "Intel SGX explained," *IACR Cryptol. ePrint Archive*, vol. 2016, No. 86, pp. 1-118, Jan 2016.

[4] Asanović, Krste, and David A. Patterson. "Instruction sets should be free: The case for risc-v." EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2014-146, Aug 2014.

[5] Patterson, David, and Andrew Waterman. The RISC-V Reader: an open architecture Atlas, *Strawberry Canyon*, pp. 3-5, 2017.

[6] Elsadek, Islam, and Eslam Yahya Tawfik. "RISC-V resource-constrained cores: A survey and energy comparison." *2021 19th IEEE International New Circuits and Systems Conference (NEWCAS)*. IEEE, June 2021. https://doi.org/10.1109/NEWCAS50681.2021.9462781

[7] Schiavone, Pasquale Davide, et al. "Slow and steady wins the race? A comparison of ultra-low-power RISC-V cores for Internet-of-Things applications." *2017 27th International Symposium on Power and Timing Modeling, Optimization and Simulation (PATMOS)*. IEEE, Sep 2017. https://doi.org/10.1109/PATMOS.2017.8106976

[8] Ibex: an embedded 32 bit RISC-V CPU core [Internet]. Available: https://ibex-core.readthedocs.io/.

[9] PMP Enhancements for memory access and execution prevention on Machine mode [Internet]. Available: https://raw.githubusercontent.com/riscv/riscv-tee/main/Smepmp/Smepmp.pdf.

[10] Hex Five Security [Internet]. Available: https://hex-five.com/.

[11] Lee, Dayeol, et al. "Keystone: An open framework for architecting trusted execution environments." *Proceedings of the Fifteenth European Conference on Computer Systems*. No. 38, pp. 1-16, Apr 2020. https://doi.org/10.1145/3342195.3387532

[12] Cheang, Kevin, et al. "Verifying RISC-V physical memory protection." *IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS) Workshop on Secure RISC-V Architecture Design*, IEEE, Aug 2020.

[13] Kumar, Vinay BY, et al. "Itus: A secure risc-v system-on-chip." *2019 32nd IEEE International System-on-Chip Conference (SOCC)*, IEEE, Sep 2019. https://doi.org/10.1109/SOCC46988.2019.1570564307

[14] Mushtaq, Muhammad Faheem, et al. "A survey on the cryptographic encryption algorithms." *International Journal of Advanced Computer Science and Applications*, Vol. 8, No. 11, pp.333-344, 2017. https://doi.org/10.14569/IJACSA.2017.081141

[15] Rogaway, Phillip. "Authenticated-encryption with associated-data." *Proceedings of the 9th ACM Conference on Computer and Communications Security*, Washington DC, pp. 98-107, Nov 2002. https://doi.org/10.1145/586110.586125

[16] Dworkin, Morris J. recommendation for block cipher modes of operation: Galois/counter mode (gcm) and gmac. National Institute of Standards & Technology, Gaithersburg, Sp 800-38d, pp. 1-39, Nov 2007.

[17] Li, Hongwei, et al. "An efficient merkle-tree-based authentication scheme for smart grid." *IEEE Systems Journal*, Vol. 8, No. 2, pp. 655-663, Jul 2013. https://doi.org/10.1109/JSYST.2013.2271537

[18] Vujičić, Dejan, Dijana Jagodić, and Siniša Ranđić. "Blockchain technology, bitcoin, and Ethereum: A brief overview." *2018 17th international symposium infoteh-jahorina (infoteh)*, IEEE, pp. 1-7, Mar 2018. https://doi.org/10.1109/INFOTEH.2018.8345547

[19] Chandra, Sourabh, et al. "A comparative survey of symmetric and asymmetric key cryptography." *2014 international conference on electronics, communication and computational engineering (ICECCE)*. IEEE, pp. 1-11, Nov 2014. https://doi.org/10.1109/ICECCE.2014.7086640

[20] McGrew, David. An interface and algorithms for authenticated encryption, IETF, Wilmington, rfc5116, Jan 2008.

[21] Krawczyk, Hugo, and Pasi Eronen. HMAC-based extract-and-expand key derivation function (HKDF), IETF, Wilmington, rfc5869, May 2010.

[22] Mukhtar, Muhammad Asim, Muhammad Khurram Bhatti, and Guy Gogniat. "Architectures for Security: A comparative analysis of hardware security features in Intel SGX and ARM TrustZone." *2019 2nd International Conference on Communication, Computing and Digital systems (C-CODE)*. IEEE, pp. 1-6, Apr 2019.

[23] Chen, Guoxing, et al. "Sgxpectre: Stealing intel secrets from sgx enclaves via speculative execution." *2019 IEEE European Symposium on Security and Privacy (EuroS&P)*, IEEE, pp 1-16, Aug 2019.

박우정(Woojung Park)

2021년 : 부산대학교 컴퓨터공학과 (공학학사)

2021년~현 재: 스마트엠투엠
※관심분야 : 하드웨어 보안

김해용(Haeyoung Kim)

2015년 : 부산대학교 전자공학과 (공학학사)
2019년 : 부산대학교 대학원 석박통합과정 수료

2022년~현 재: 스마트엠투엠
※관심분야 : IoT, 하드웨어 보안

지장현(Janghyun Ji)

2016년 : 부산대학교 컴퓨터공학과 (공학학사)
2021년 : 부산대학교 대학원 석박통합과정 수료

2020년~현 재: 스마트엠투엠
※관심분야 : 정보보호, 하드웨어 보안

문재근(Jaegeun Moon)

2015년 : 경북대학교 전자공학과 (공학학사)
2017년 : 경북대학교 대학원 (공학석사)

2017년~2018년: 한국정보통신기술협회
2021년~현 재: 스마트엠투엠
※관심분야 : 정보보호, 하드웨어 보안, 부채널 분석

김호원(Howon Kim)

1995년 : 포항공과대학교 대학원 공학 석사
1999년 : 포항공과대학교 대학원 공학 박사
2004년 : Ruhr University Bochum, Post Doctorial

1998년~2008년: 한국전자통신연구원 팀장
2008년~현 재: 부산대학교 전기컴퓨터공학부 교수
2020년~현 재: 스마트엠투엠 연구소장
※관심분야 : 정보보호, IoT, 인공지능, 블록체인